Homework 3

EECS 270 Winter 2020

Due Friday, February 7 @ <u>11:59 PM</u> on Gradescope

This is an individual assignment, all of the work should be your own.
Write neatly or type and show all your work for full credit.
Have your name and unique name on the front page of your submission.

1. **[20 points]** Let $F = (x'z+y)(x \bigoplus z)+(yz')(x+z')$. Repeatedly apply Shannon's Expansion Theorem to put F into canonical SOP form. *hint: Remember that repeatedly applying Shannon's Expansion on a function gets you to* $a \cdot b + b \cdot c$ *canonical forms. On this problem, expand once, simplify the co-factors (like in HW 2 Q2). Then expand the other variables until you get to a canonical form. The key to each step is simplifying the co-factors* ***only****, until you've reached the point you can simplify the entire function into the canonical SOP form.*

2. **[10 points]** Switching Algebra:

   a. **[5]** Is the following property true or false? $a + (b \bigoplus c) = (a + b) \bigoplus (a + c)$. If it is true, prove it algebraically. If it is false, show the work you did to determine it is not equal & provide a counterexample.

   b. **[5]** Determine if the set of operations {XOR, OR} is functionally complete and explain why.

3. **[30 points]** Design a circuit with the following description:

   A network router connects multiple computers together and allows them to send messages to each other. If two or more computers send messages simultaneously, the messages "collide" and the messages must be resent. Using the combinational design process, create a collision detection circuit for a router that connects 4 computers. The circuit has 4 inputs labeled M_0 through M_3 that are 1 when the corresponding computer is sending a message and 0 otherwise. The collision output C is 1 when a collision is detected and 0 otherwise.

   In addition, 3 computers on the network exhibit frequent activity: M_3, M_2, and M_0. Because of this, an additional output, H controls a high-throughput bus. This bus is expensive to operate, so it's only enabled if *any combination of* 2 of these machines or all 3 of the machines are sending messages at once.

   Finally, the detection circuit alerts an administrator if there's excessive traffic, *i.e.*, all machines transmitting. This outputs an additional signal, A.

   a. **[10]** Create a truth table for all possible input combinations with a column for each of the outputs.

   b. **[14]** Write SOP equations for each of the outputs (note that they do <u>not</u> have to be **canonical SOP**).

   c. **[6]** Draw a gate-level logic circuit for each of the outputs using your equations from (b). Your gates can be any size input you need (*e.g., 2-input, 3-input, etc.*), except for inverters which are always 1-input.

4. **[20 points]** Design a circuit with the following description:

   Congratulations, you've just finished constructing your new Minecraft dirt home! However, you need to protect it from your arch-nemeses. Luckily, your best friend Bagely offers a suggestion that you build a coded door lock:

   You have 6 switches to control your door, from I_5 - I_0. These inputs represent an unsigned binary number $I_5I_4I_3I_2I_1I_0$ that encodes your secret code. If the secret code is 42_{10}, or the binary value 101010_2, the door will open. As a fail-safe, you've also hidden another switch that acts as an emergency override,

Homework 3

EECS 270 Winter 2020

Due Friday, February 7 @ <u>**11:59 PM**</u> on Gradescope

This is an individual assignment, all of the work should be your own.
Write neatly or type and show all your work for full credit.
Have your name and unique name on the front page of your submission.

1. **[20 points]** Let $F = (x'z+y)(x \bigoplus z)+(yz')(x+z')$. Repeatedly apply Shannon's Expansion Theorem to put F into canonical SOP form. *hint: Remember that repeatedly applying Shannon's Expansion on a function gets you to canonical forms. On this problem, expand once, simplify the co-factors (like in HW 2 Q2). Then expand the other variables until you get to a canonical form. The key to each step is simplifying the co-factors* ***only****, until you've reached the point you can simplify the entire function into the canonical SOP form.*

$F(x,y,z) = x'F(0,y,z) + xF(1,y,z)$

$F(0,y,z) = (z+y)(0 \oplus z) + (yz')z'$

$= (z+y)(0'z) + (yz')z'$

$= zz + yz + yz'z'$

$= z + yz + yz'$

$= z + yz'$

$= z + y.$

$F(1,y,z) = (y)(1 \oplus z) + (yz')$

$= y(z') + yz'$

$= yz' + yz'$

$= yz'$

$= x'(z+y) + x(yz')$

$= x'z + x'y + xyz'$

$= x'z(y+y') + x'y(z+z') + xyz'$

$= x'zy + x'zy' + x'yz + x'yz' + xyz'$

P $= \underline{x'yz} + \underline{x'y'z} + \cancel{x'yz} + \underline{x'yz'} + \underline{xyz'}$

Homework 3

EECS 270 Winter 2020

Due Friday, February 7 @ <u>**11:59 PM**</u> on Gradescope

This is an individual assignment, all of the work should be your own.
Write neatly or type and show all your work for full credit.
Have your name and unique name on the front page of your submission.

1. **[20 points]** Let $F = (x'z+y)(x \bigoplus z)+(yz')(x+z')$. Repeatedly apply Shannon's Expansion Theorem to put F into canonical SOP form. *hint: Remember that repeatedly applying Shannon's Expansion on a function gets you to canonical forms. On this problem, expand once, simplify the co-factors (like in HW 2 Q2). Then expand the other variables until you get to a canonical form. The key to each step is simplifying the co-factors* ***only****, until you've reached the point you can simplify the entire function into the canonical SOP form.*

2. **[10 points]** Switching Algebra:

   a. **[5]** Is the following property true or false? $a + (b \bigoplus c) = (a + b) \bigoplus (a + c)$. If it is true, prove it algebraically. If it is false, show the work you did to determine it is not equal & provide a counterexample.

   b. **[5]** Determine if the set of operations {XOR, OR} is functionally complete and explain why.

a) LHS:

$a+(b \oplus c) = (a+b) \oplus (a+c).$

$a + b'c + bc'$

RHS:

$(a+b)'(a+c) + (a+b)(a+c)'$

$(a'b')(a+c) + (a+b)(a'c')$

$a'ab' + a'b'c + aa'c' + ba'c'$

$0 + a'b'c + 0 + a'bc'$

$a'b'c + a'bc'$

$a'(b'c + bc')$

$a'(b \oplus c)$

LHS ≠ RHS.

Let $a = 1$ $b = 0$ $c = 0$

$LHS = 1 + 0 + 0 = 0.$

$RHS = 0 \cdot 0 = 0.$

XOR $\oplus$ OR $+$.

{NOT. or . and . Nor}.

By definition. {and . or. not}.

$A \oplus B = A'B + AB'$

Let $B = 1$

$A \oplus 1 = A'$ ← we got not here.

$(A' + B')' = AB$ ← and.

3. **[30 points]** Design a circuit with the following description:

A network router connects multiple computers together and allows them to send messages to each other. If two or more computers send messages simultaneously, the messages "collide" and the messages must be resent. Using the combinational design process, create a collision detection circuit for a router that connects 4 computers. The circuit has 4 inputs labeled M_0 through M_3 that are 1 when the corresponding computer is sending a message and 0 otherwise. The collision output C is 1 when a collision is detected and 0 otherwise.

In addition, 3 computers on the network exhibit frequent activity: M_3, M_2, and M_0. Because of this, an additional output, H controls a high-throughput bus. This bus is expensive to operate, so it's only enabled if *any combination of* 2 of these machines or all 3 of the machines are sending messages at once.

Finally, the detection circuit alerts an administrator if there's excessive traffic, *i.e.*, all machines transmitting. This outputs an additional signal, A.

a. **[10]** Create a truth table for all possible input combinations with a column for each of the outputs.

b. **[14]** Write SOP equations for each of the outputs (note that they do not have to be **canonical SOP**).

c. **[6]** Draw a gate-level logic circuit for each of the outputs using your equations from (b). Your gates can be any size input you need (*e.g., 2-input, 3-input, etc.*), except for inverters which are always 1-input.

b) $C = \underline{M_3'M_2'M_1'M_0' + M_3'M_2'M_1'M_0} + M_3'M_2'M_1M_0'$

$+ M_3'M_2M_1'M_0' + M_3M_2'M_1'M_0'$

$M_3'M_2'M_1' + M_3'M_0'(M_2'M_1 + M_2M_1') + M_3M_2'M_1'M_0'$

$M_2'M_1'(M_3' + M_3M_0')$

$$\begin{array}{cccccc} 1 & 0 & 1 & 0 & 1 & 0 \\ \hline 5 & 4 & 3 & 2 & 1 & 0 \end{array}$$

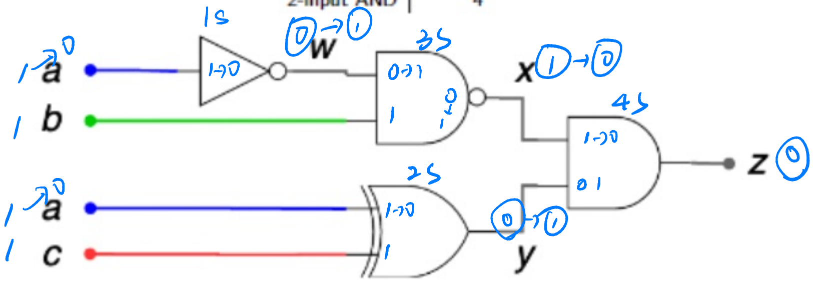
$O = E + I_5 I_4' I_3 I_2' I_1 I_0'$

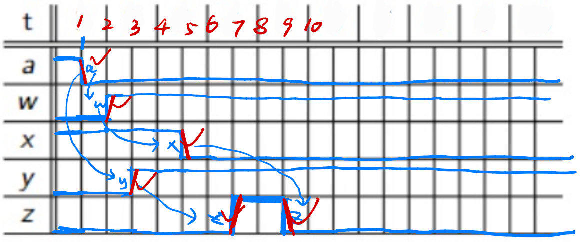
668 $\begin{array}{cccccc} 1 & 1 & 0 & 1 & 1 & 0 \\ \hline 5 & 4 & 3 & 2 & 1 & 0 \end{array}$

$S = I_5 I_4 I_3' I_2 I_1 I_0'$

5) 5. **[20 points]** Draw a timing diagram with causality arrows for a, w, x, y, and z for the following circuit and associated delays.
Assume that a, b, and c have all been 1 for a long time. At t=1, a transitions from 1 → 0.
Want a program to draw timing diagrams? Some of the ones out there: WaveDrom and Waveme. You're also free to draw freehand or other methods you may find!

Gate	Delay (units)
NOT	1
2-input XOR	2
2-input NAND	3
2-input AND	4





E, that will open the door *regardless of the state of the switches I_5 - I_0*. The signal to open the door is *O*.

However, you're extremely clever and foresee being asked for the door code. You design in a dummy code such that if 66_8 is entered, the self-destruct signal, *S*, will trigger a massive TNT explosion (please do be careful with that one).

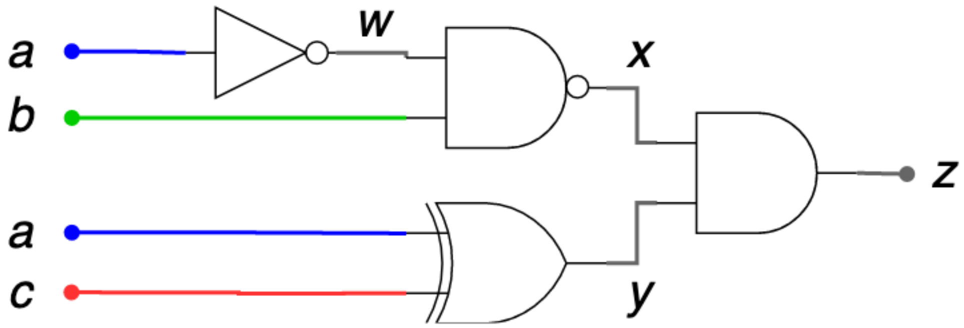
a. **[5]** Should you use a truth table here to help you design your circuit? Why or why not?

b. **[10]** Regardless of your answer to (a), design SOP equations for each of the outputs (note that they do not have to be **canonical SOP**).

c. **[5]** Draw a gate-level logic circuit for each of the outputs using your equations from (b). RESTRICTION: You must do this using only 2-input devices, except an inverter is 1-input of course. In other words, an AND gate can only accept 2 signals to be AND'd together.

5. **[20 points]** Draw a timing diagram with causality arrows for *a*, *w*, *x*, *y*, and *z* for the following circuit and associated delays.
Assume that *a*, *b*, and *c* have all been 1 for a long time. At t=1, *a* transitions from 1 → 0.
Want a program to draw timing diagrams? Some of the ones out there: WaveDrom and Waveme. You're also free to draw freehand or other methods you may find!

Gate	Delay (units)
NOT	1
2-input XOR	2
2-input NAND	3
2-input AND	4



t	1	2	3	4	5	6	7	8	9	10
a										
w										
x										
y										
z										